

# Road to Confidence in IT Systems: SAMATE's SATE and SRD projects

Paul E. Black

paul.black@nist.gov



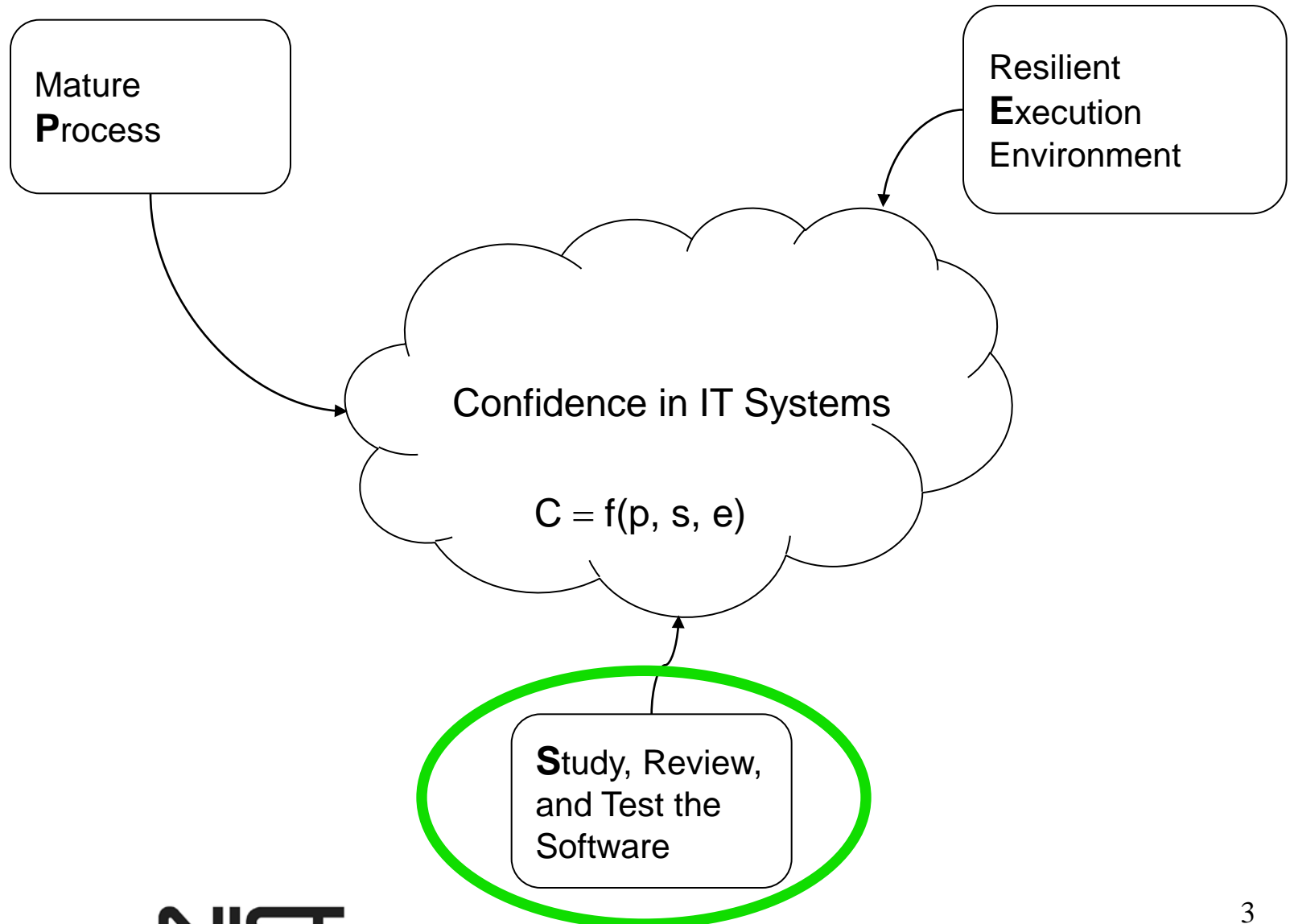
# Software Assurance Metrics And Tool Evaluation (SAMATE) project

## Current Areas of Concentration

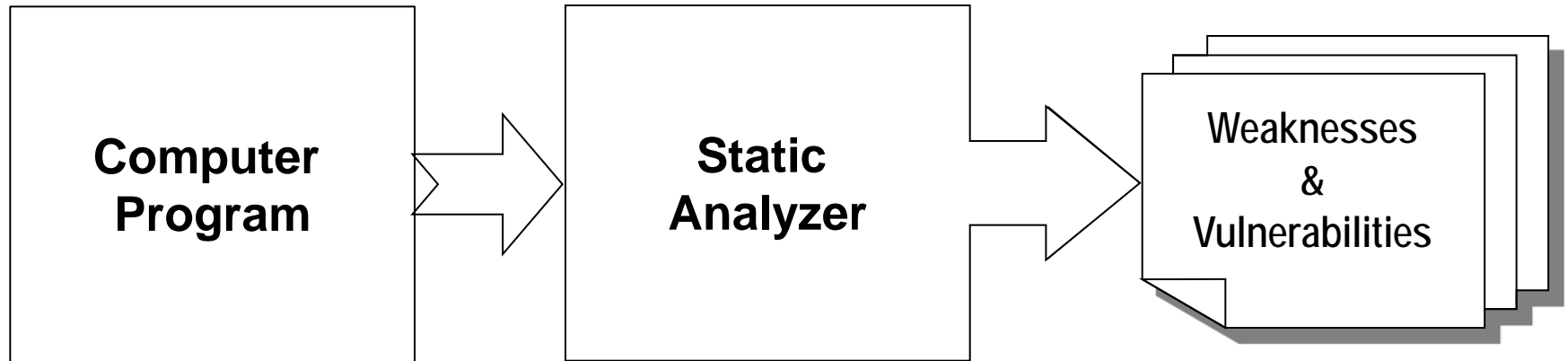
- **Static source code security analyzers**
- **Static Analysis Tool Exposition (SATE)**
- **SAMATE Reference Dataset (SRD)**
- **Vote tools evaluation methodology**
- **Studies**

**<http://samate.nist.gov/>**

# SAMATE's Niche



# What is a Static Analyzer?



- **Static analyzers check programs for bugs.**
- **They can't catch everything**
  - e.g. sending data: should it be encrypted?

# How Good are Static Analyzers?

- **Does this analyzer find all bugs?**
  - without too many false positives ...
- **Does it find all the bugs we know about?**
- **How well does it do for important bugs?**
- **How much is my confidence increased by using it?**
- **Is running it a good use of my time?**

# Static Analysis Tool Exposition

## SATE

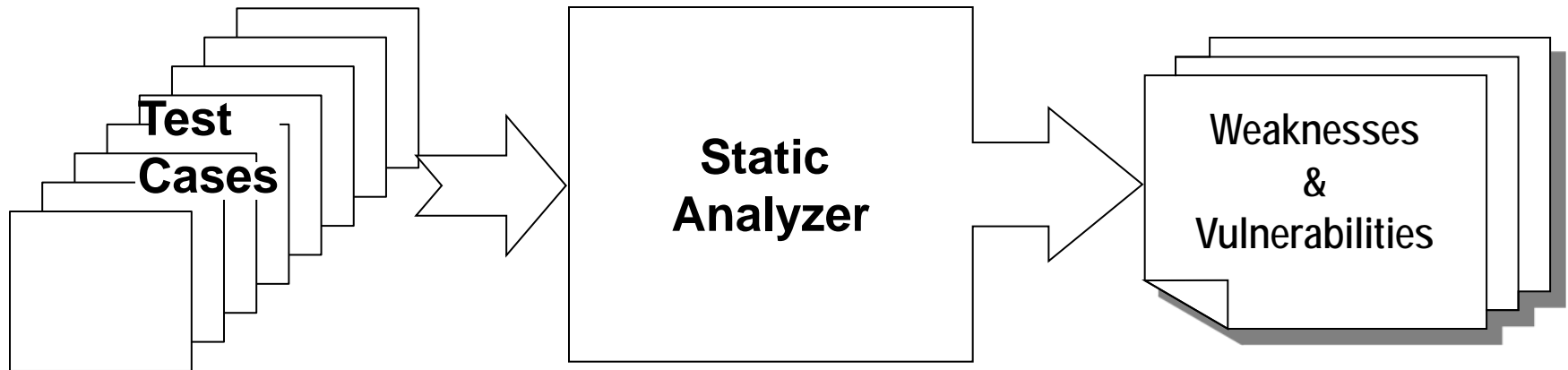
- **Goals:**
  - Gather large test sets to enable empirical research
  - Encourage improvement of tools
  - Speed adoption of tools by objectively demonstrating their use on real software
- **Steps**
  - We choose open source programs with security implications
  - Participants run tools and return reports
  - We analyze reports
  - Everyone shares observations at a workshop
  - We release final reports and data later

<http://samate.nist.gov/SATE.html>

# SATE Overview

- **We held four SATEs since 2008.**
- **Test sets were in C/C++, Java, and PHP and consisted of about 26,000,000 LoC.**
- **19 teams on 4 continents participated.**
- **We received a combined 128,043 warnings\***
- **We better understand subtleties of finding, designating, and counting weaknesses.**

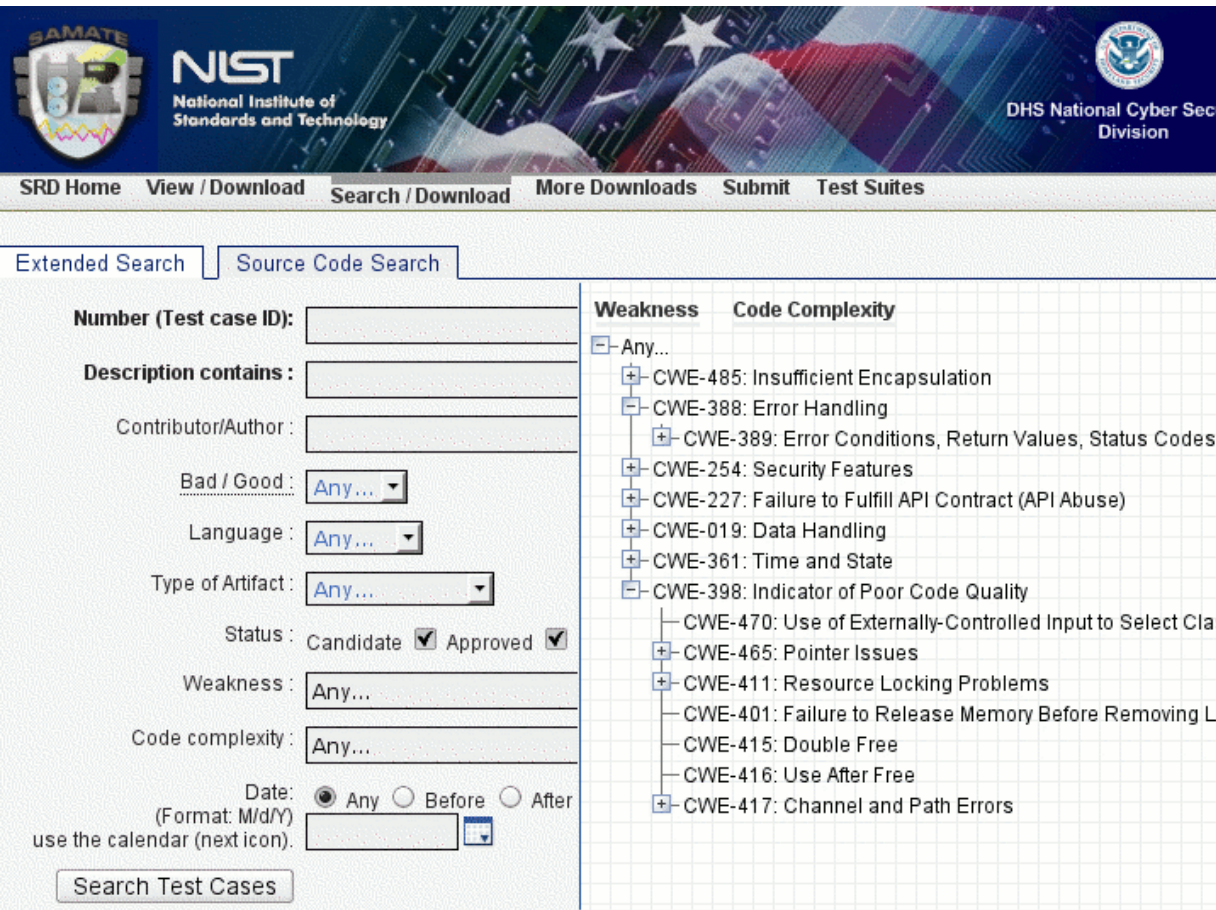
# How can We Test Static Analyzers?



- **Come up with a small set of test programs with known bugs.**
- **The result of running a tool on the set is correlated with the result on large, real-world programs.**



# SAMATE Reference Dataset (SRD)




The screenshot shows the SAMATE SRD search interface. At the top, there is a banner with the SAMATE logo, NIST logo, and DHS National Cyber Security Division logo. Below the banner is a navigation bar with links: SRD Home, View / Download, Search / Download, More Downloads, Submit, and Test Suites. The main search area has two tabs: Extended Search and Source Code Search. The Source Code Search tab is active. On the left, there are search filters: Number (Test case ID), Description contains, Contributor/Author, Bad / Good (Any...), Language (Any...), Type of Artifact (Any...), Status (Candidate, Approved), Weakness (Any...), Code complexity (Any...), and Date (Any, Before, After). A 'Search Test Cases' button is at the bottom left. On the right, there is a tree view of weaknesses and code complexity categories. The tree view shows a hierarchy of weaknesses, including CWE-485: Insufficient Encapsulation, CWE-388: Error Handling, CWE-389: Error Conditions, Return Values, Status Codes, CWE-254: Security Features, CWE-227: Failure to Fulfill API Contract (API Abuse), CWE-019: Data Handling, CWE-361: Time and State, CWE-398: Indicator of Poor Code Quality, CWE-470: Use of Externally-Controlled Input to Select Class, CWE-465: Pointer Issues, CWE-411: Resource Locking Problems, CWE-401: Failure to Release Memory Before Removing Link, CWE-415: Double Free, CWE-416: Use After Free, and CWE-417: Channel and Path Errors.

- Public repository of over 60,000 test cases in C++, C, Java, C#, PHP, etc. covering 147 classes of weaknesses.

- User can search by language, weakness, code construct, etc.

- Contributions from Fortify, Defence R&D Canada, Klocwork, MIT Lincoln Laboratory, Praxis, Secure Software, and others.

[samate.nist.gov/SRD](http://samate.nist.gov/SRD)

58	2005-11-02	Java	Source Code	SecureSoftware	C	Not using a a random initialization vector with Cipher Block ...	
71	2005-11-07	Java	Source Code	SecureSoftware	C	Omitting a break statement so that one may fall through is often ...	
1552	2006-06-22	Java	Source Code	Jeff Meister	C	Tainted input allows arbitrary files to be read and written.	
1553	2006-06-22	Java	Source Code	Jeff Meister	C	Tainted input allows arbitrary files to be read and written. ...	
1554	2006-06-22	Java	Source Code	Jeff Meister	C	Two file operations are performed on a filename, allowing a filenameer	
1567	2006-06-22	Java	Source Code	Jeff Meister	C	The credentials for connecting to the database are hard-wired ...	
1568	2006-06-22	Java	Source Code	Jeff Meister	C	The credentials for connecting to the database are hard-wired ...	
1569	2006-06-22	Java	Source Code	Jeff Meister	C	The credentials for connecting to the database are hard-wired ...	
1570	2006-06-22	Java	Source Code	Jeff Meister	C	An exception leaks internal path information to the user.	
1571	2006-06-22	Java	Source Code	Jeff Meister	C	An exception leaks internal path information to the user. (fixed ...	
1579	2006-06-22	Java	Source Code	Jeff Meister	C	Tainted output allows log entries to be forged.	

```

public class File1_bad extends HttpServlet
{
    public void doGet(HttpServletRequest req, HttpServletResponse res)
        throws ServletException, IOException
    {
        res.setContentType("text/html");
        ServletOutputStream out = res.getOutputStream();
        out.println("<HTML><HEAD><TITLE>Test</TITLE></HEAD><BODY><blockquote><pre>");

        String name = req.getParameter("name");
        String msg = req.getParameter("msg");
        if(name != null) {
            try {
                File f = new File("/tmp", name);           /* BAD */
                if(msg != null) {
                    FileWriter fw = new FileWriter(f);      /* BAD */
                    fw.write(msg, 0, msg.length());
                    fw.close();
                    out.println("message stored");
                } else {
                    String line;
                    BufferedReader fr = new BufferedReader(new FileReader(f));
                    while((line = fr.readLine()) != null)
                        out.println(line);
                }
            } catch(Exception e) {
                throw new ServletException(e);
            }
        } else {

```